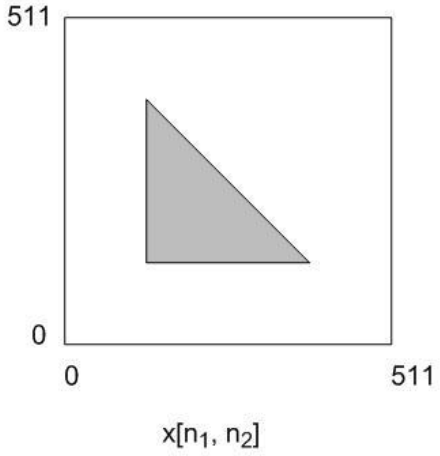
Stevens Institute of Technology
Department of Electrical and Computer Engineering

Spring Semester 2025

CpE 462 Introduction to Image Processing

Homework 7: Due Apr. 24

7.1 Assume that a bi-level input image $\mathbf{x[n_1, n_2]}$ of **512×512** as shown below, where the dark region has amplitude of **50** and the white background has amplitude of **200**.



An edge detector (consisting of two filters) is applied to $\mathbf{x[n_1, n_2]}$ which will produce two filtered images $\mathbf{G_1(x[n_1, n_2])}$ and $\mathbf{G_2(x[n_1, n_2])}$. These two images will be combined to form one gray level image $\mathbf{y[n_1, n_2]}$ using the absolute sum

$$\mathbf{y[n_1, n_2]=|G_1(x[n_1, n_2])| + |G_2(x[n_1, n_2])|}$$

A segmentation will then be performed on this image to produce a binary edge image. Assume that segmentation is based on the operator

$$z[n_1,n_2]=\begin{cases}255\ (\textit{white}) & \textit{if} \quad y[n_1,n_2]\geq T\\ 0\ (\textit{black}) & \textit{if} \quad y[n_1,n_2]<T\end{cases}$$

7.1.1. Use Roberts edge detector on this image. Select an appropriate threshold values **T** such that **z[n1, n2]** will only show all the edges. Specify your threshold **T** and sketch your output image.

7.1.2. Repeat **7.1.1.** using Prewitt edge detector.

(**Note:** each actual edge may result in single, double, triple or more lines depending on different edge detector, please specify the lines you get.)

7.2 Based on the **imageprocessing.c** structure, write a small routine which can automatically calculate the global threshold value according to the iterative global threshold estimation algorithm we discussed in class.

Hint: you have to initialize a **T**; then read through the image several times to update the **T**; your iteration will stop when your newly updated $\mathbf{T_i}$ is not much different from the previous $\mathbf{T_{i-1}}$, i.e. $|\mathbf{T_i - T_{i-1}}| < \mathbf{a}$. You can let $\mathbf{a = 5}$ for example. You should try to let the program display the updated $\mathbf{T_i}$ at each iteration so you'll have an idea of whether it is running properly. Finally you should apply your final **T** to the image and obtained a binary output image and print out the result.

$$\mathbf{image\_out[n_1, n_2]} = \begin{cases} \mathbf{255\ (white)} & \mathbf{if} \quad \mathbf{image\_in[n_1, n_2] \geq T} \\ \mathbf{0\ (black)} & \mathbf{if} \quad \mathbf{image\_in[n_1, n_2] < T} \end{cases}$$